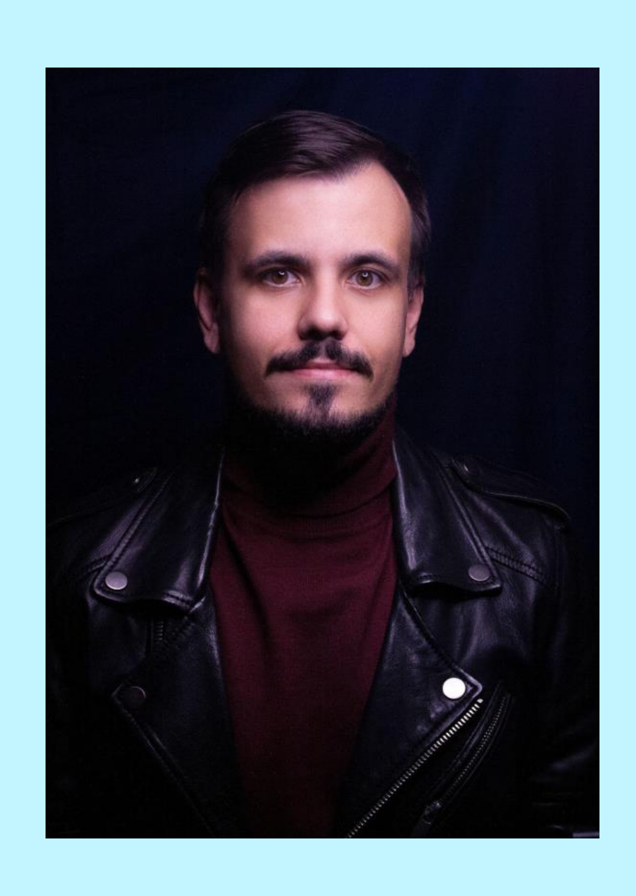

Анализ и приоритизация задач в тикетной системе
Олег Мифле
php
PHP Russia
2022



Олег Мифле

Senior backend developer
в Skypro от Skyeng

> 10 лет занимаюсь разработкой

> 7 лет пишу на PHP

➔ https://t.me/mifleo
➔ https://github.com/olegmifle
➔ mifleov@gmail.com

О чём пойдёт речь



Пара слов
про сервис

О чём пойдёт речь



Пара слов
про сервис



Погружение
в предметную
область

О чём пойдёт речь



Пара слов
про сервис

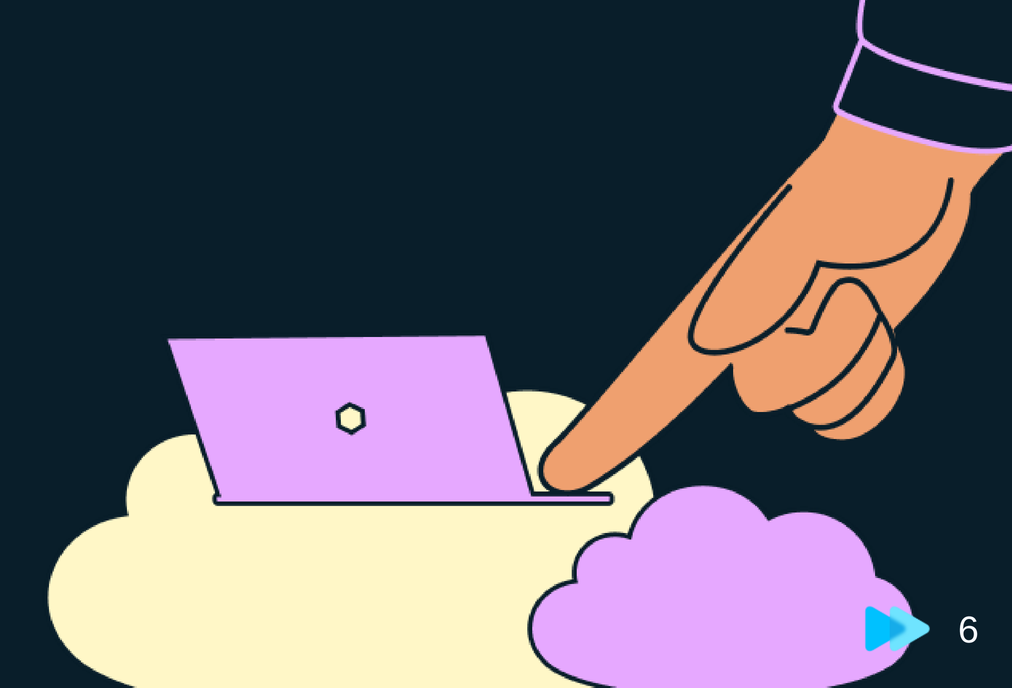


Погружение
в предметную
область



Анализ
и приоритизация

Customer Support Service



Как это было «до»?



Как это было «до»?

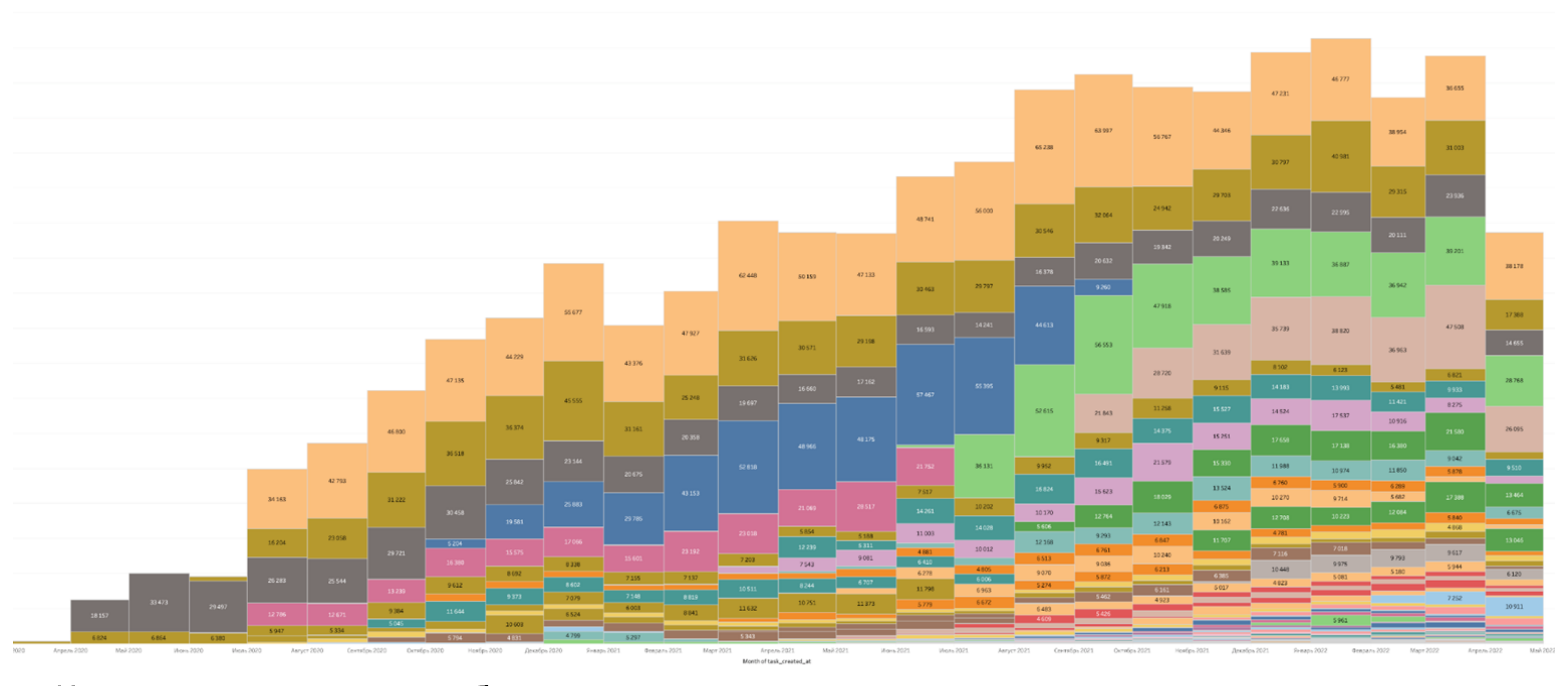
- Использовали Google Sheets, Usedesk, Omnidesk
- Все задачи ставились вручную операторами системы
- Отсутствие гибкой системы приоритетов
- Сложность построения прозрачной аналитики
- Сложности с разделением доступа



Зачем делать своё?

- Всё чаще упирались в ограничения по возможностям
- Бэклог вендора на over много месяцев
- Лимит по производительности





Как развивался и масштабировался сервис.
Разные группы операторов выделены отдельными цветами.

Customer Support Service

> 7,5 M
обработанных задач

Customer Support Service

> 7,5 M
обработанных задач

> 5 000 операторов

Customer Support Service

> 7,5 M
обработанных задач

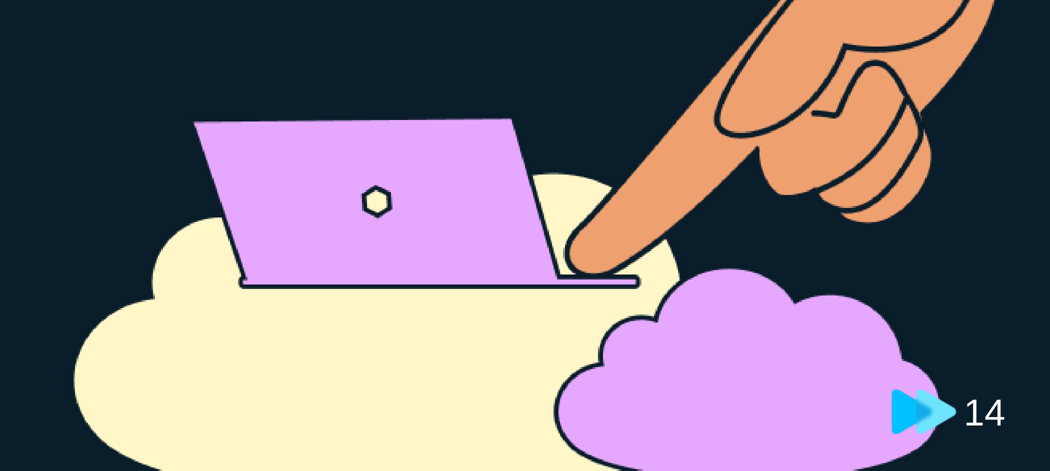
> 5 000 операторов

~ 100 задач в день
~ 350 000 в месяц
(5 минут на задачу)*

* Распределение задач на группу и оператора нелинейное.



Предметная область



Основные сущности

- Оператор и группа операторов
- Вид (шаблон) задачи
- Задача (тикет)

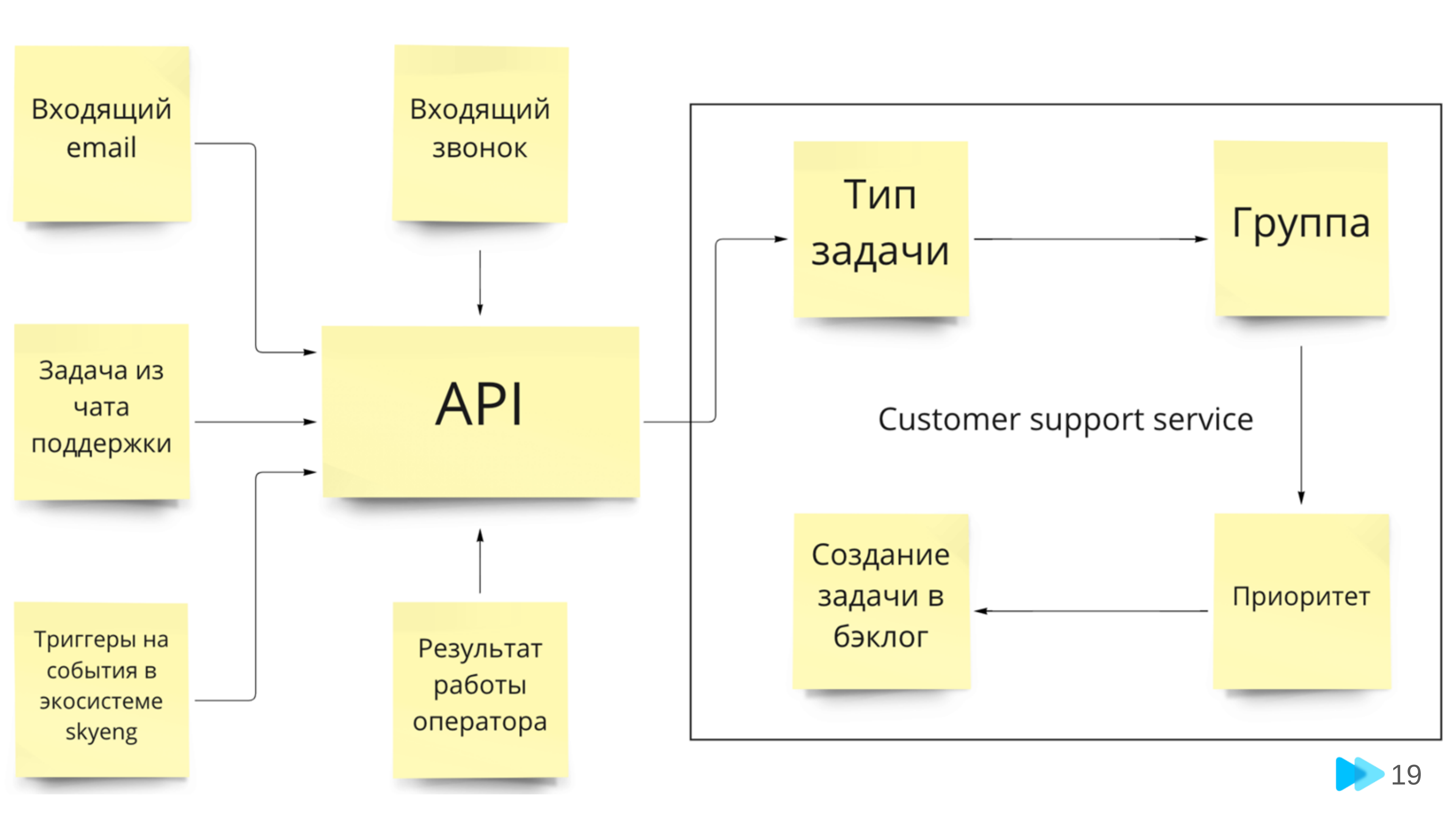
Задача

- Время на обработку задачи
- Общее время на выполнение задачи
- Конфигурационные опции
- Резолюции
- Метки

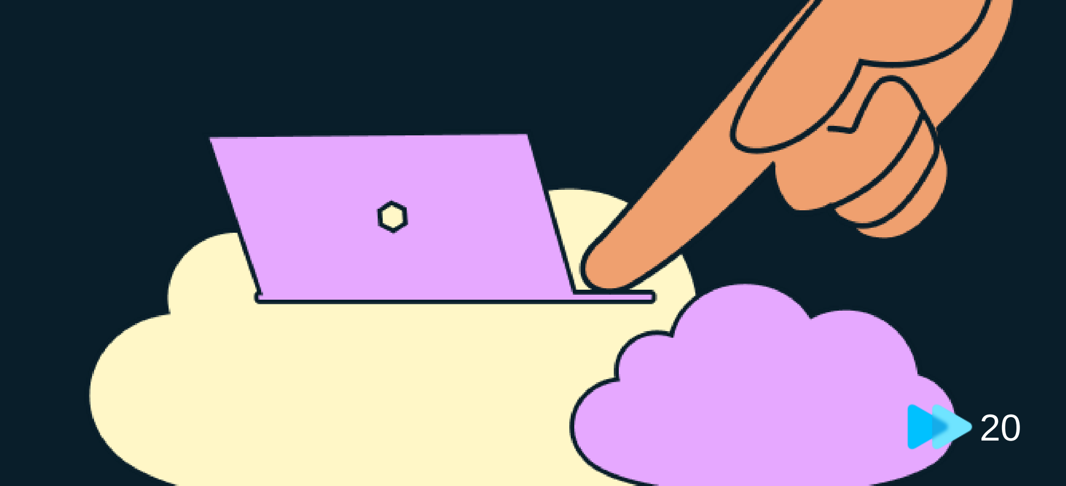
Откуда взялись все эти задачи?

Откуда берутся задачи

- Создаются при личном обращении пользователя
- Результат работы оператора над другой задачей
- Триггерные задачи – реакция на события в системе


Входящий email
Входящий звонок
Задача из чата поддержки
Триггеры на события в экосистеме skyeng
API
Результат работы оператора
Тип задачи
Группа
Customer support service
Создание задачи в бэклог
Приоритет

Приоритизация



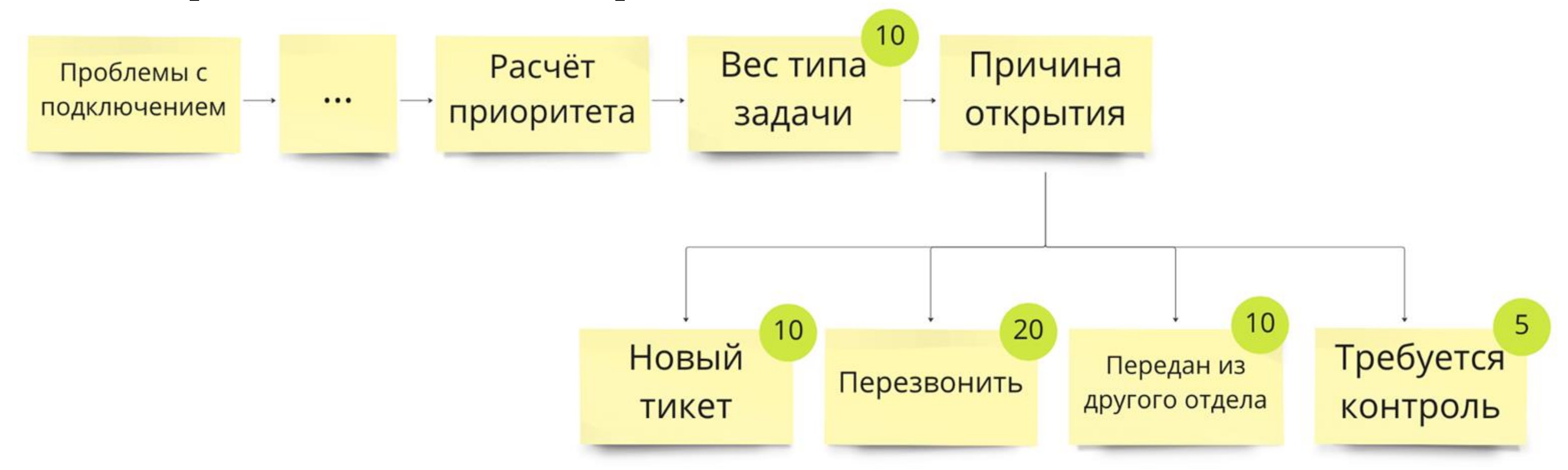
Версия первая

```
SELECT *
FROM task
ORDER BY type_id, created_at DESC
LIMIT 1
```

Версия первая

```
SELECT *
FROM task
WHERE ((operator_id IS NULL) OR (operator_id = ? AND completed_at IS
NULL))
 AND student_id NOT IN (
   SELECT student_id
   FROM task
   WHERE completed_at IS NULL
     AND operator_id IS NOT NULL
     AND operator_id <> ?
   )
ORDER BY type_id, created_at DESC
LIMIT 1
```

Версия вторая



Версия первая

Проблемы с подключением → ... → Расчёт приоритета → Вес типа задачи (10) → Причина открытия

Причина открытия:
- Новый тикет (10)
- Перезвонить (20)
- Передан из другого отдела (10)
- Требуется контроль (5)

Формула: Тип задачи + вес резолюции



Теория игр

Разновидности

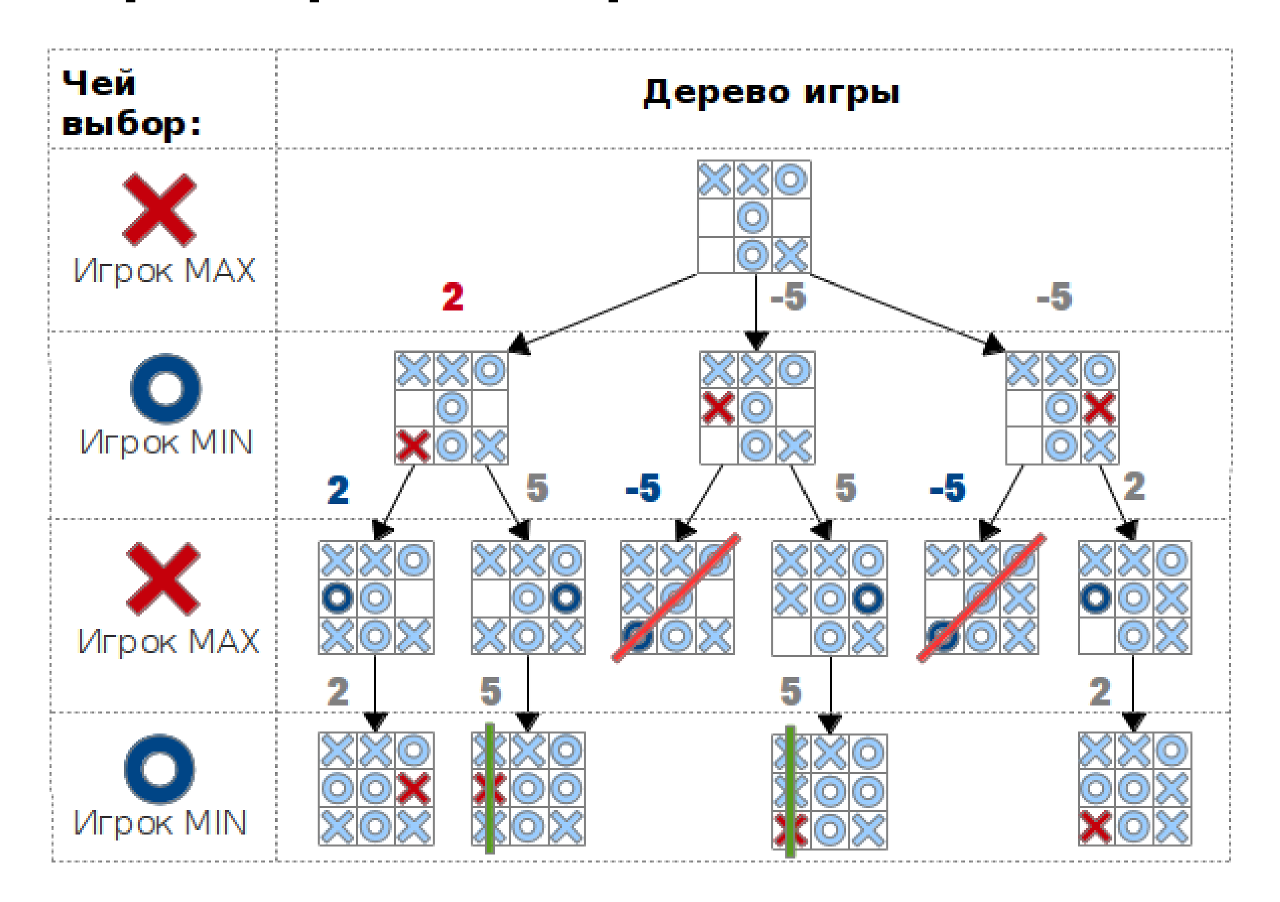
- Игры с полной информацией
- С неполной информацией

Основные понятия

- Правила игры
- Игроки
- Начальная позиция
- Позиция игры
- Партия игры
- Ход игры изменяет позицию
- Заключительная позиция



Граф игры в крестики-нолики

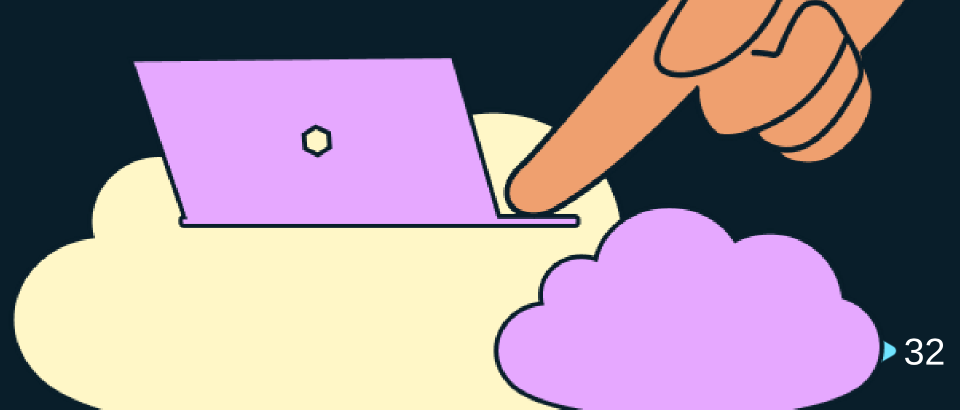


У нас нет игроков

Задачи конкурируют между собой

В конечном счёте
мы должны пройти
все «ветви»

Анализ приоритетов задач

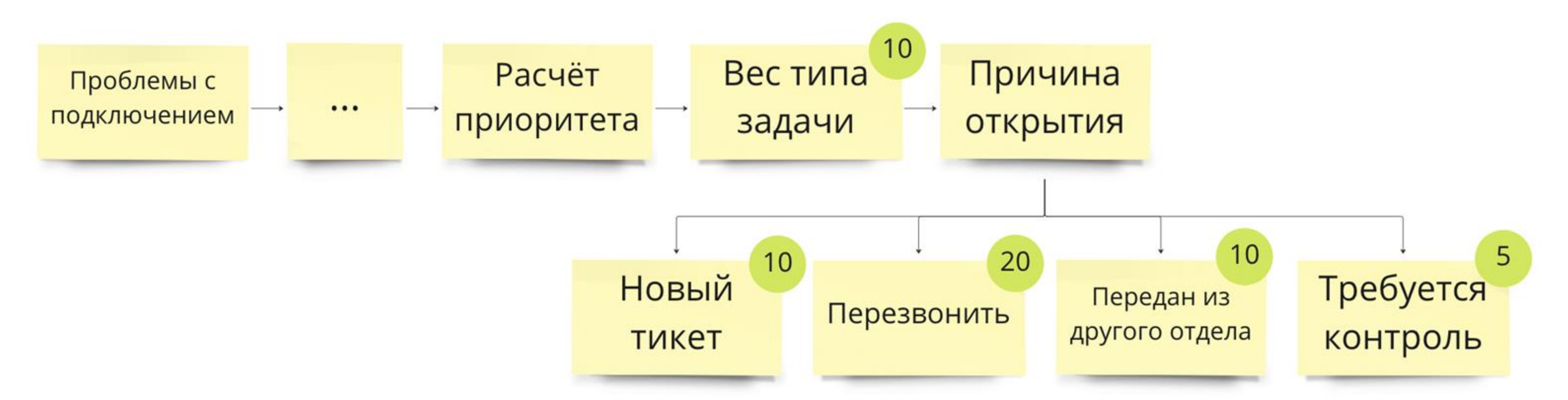


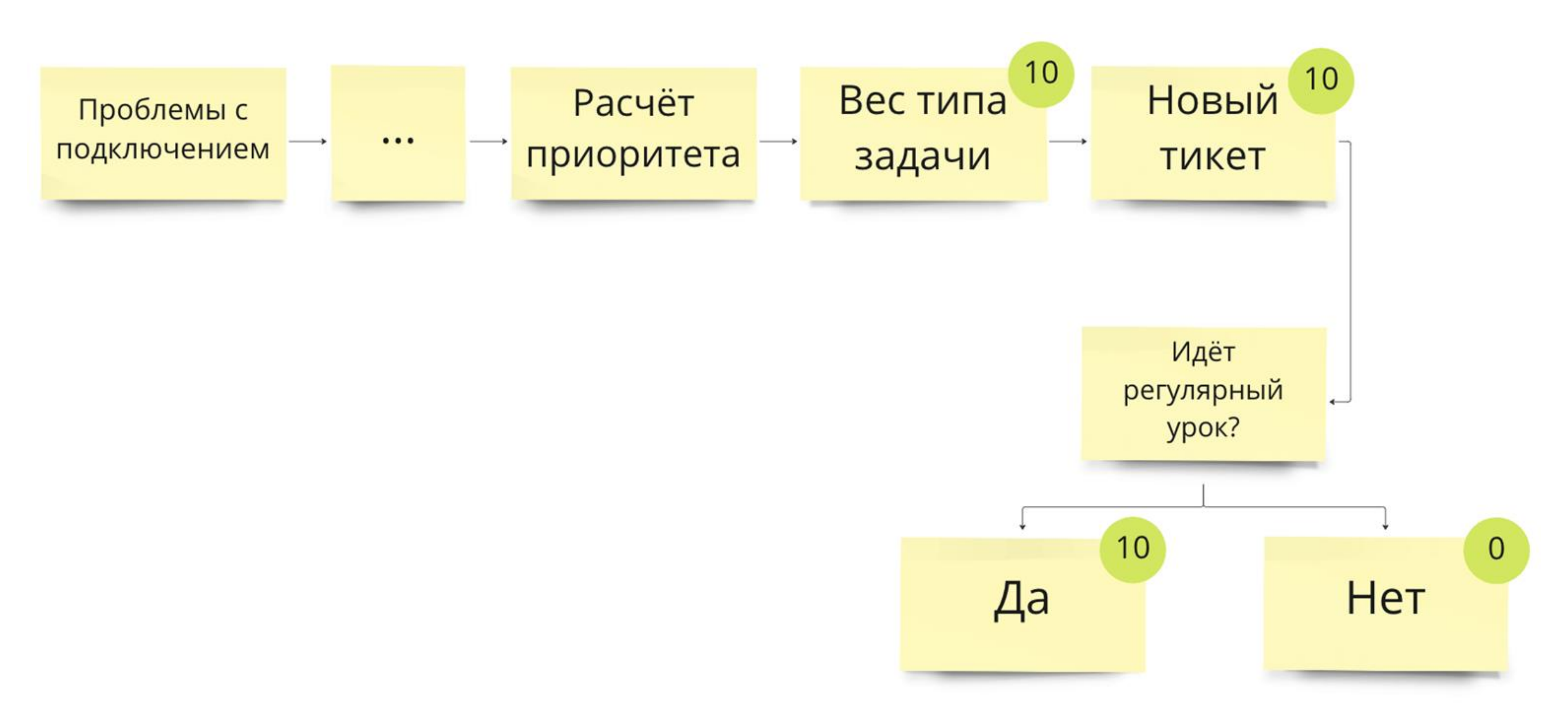
Стратегия

- У каждой группы операторов есть свои «критерии» важности

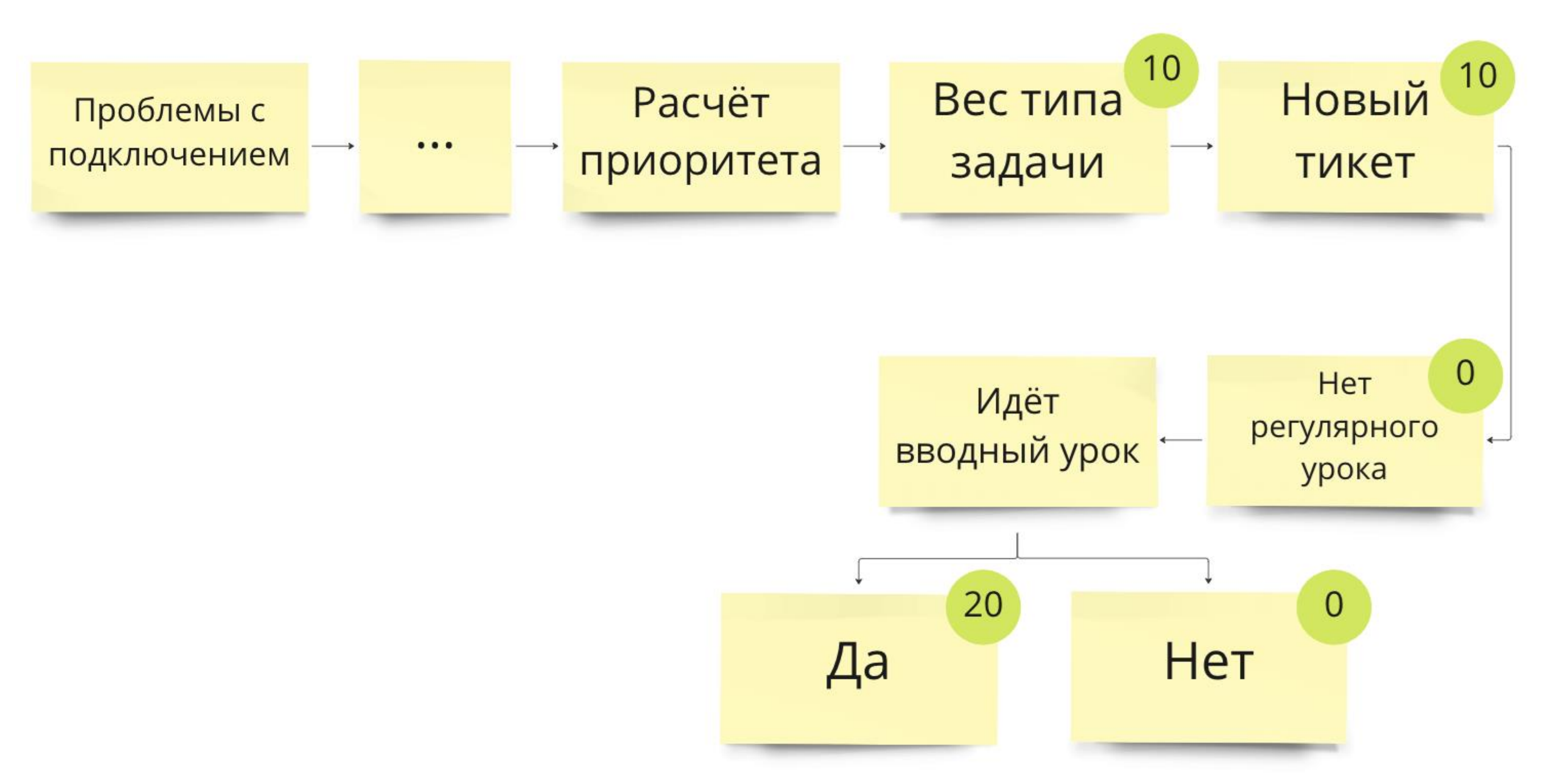
Стратегия

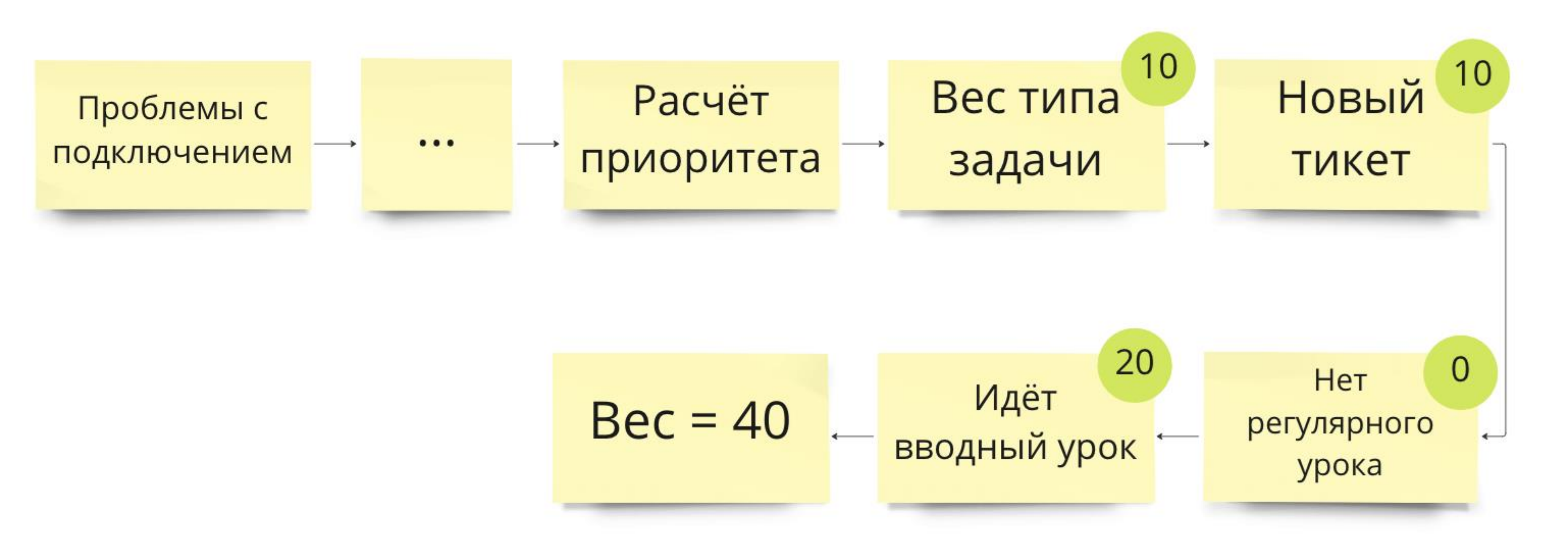
- У каждой группы операторов есть свои «критерии» важности
- Выделяем эти критерии и объединяем в стратегии расчётов


Проблемы с подключением
...
Расчёт приоритета
Вес типа задачи
10
Причина открытия
Новый тикет
10
Перезвонить
20
Передан из другого отдела
10
Требуется контроль
5



Проблемы с подключением
...
Расчёт приоритета
Вес типа задачи
10
Новый тикет
10
Идёт регулярный урок?
Да
10
Нет
0


Проблемы с подключением
...
Расчёт приоритета
Вес типа задачи
10
Новый тикет
10
Нет регулярного урока
0
Идёт вводный урок
Да
20
Нет
0


Проблемы с подключением
...
Расчёт приоритета
Вес типа задачи
10
Новый тикет
10
Нет регулярного урока
0
Идёт вводный урок
20
Вес = 40

Формула вычисления приоритета

Вес задачи = W1 + W2 + … + Wn,

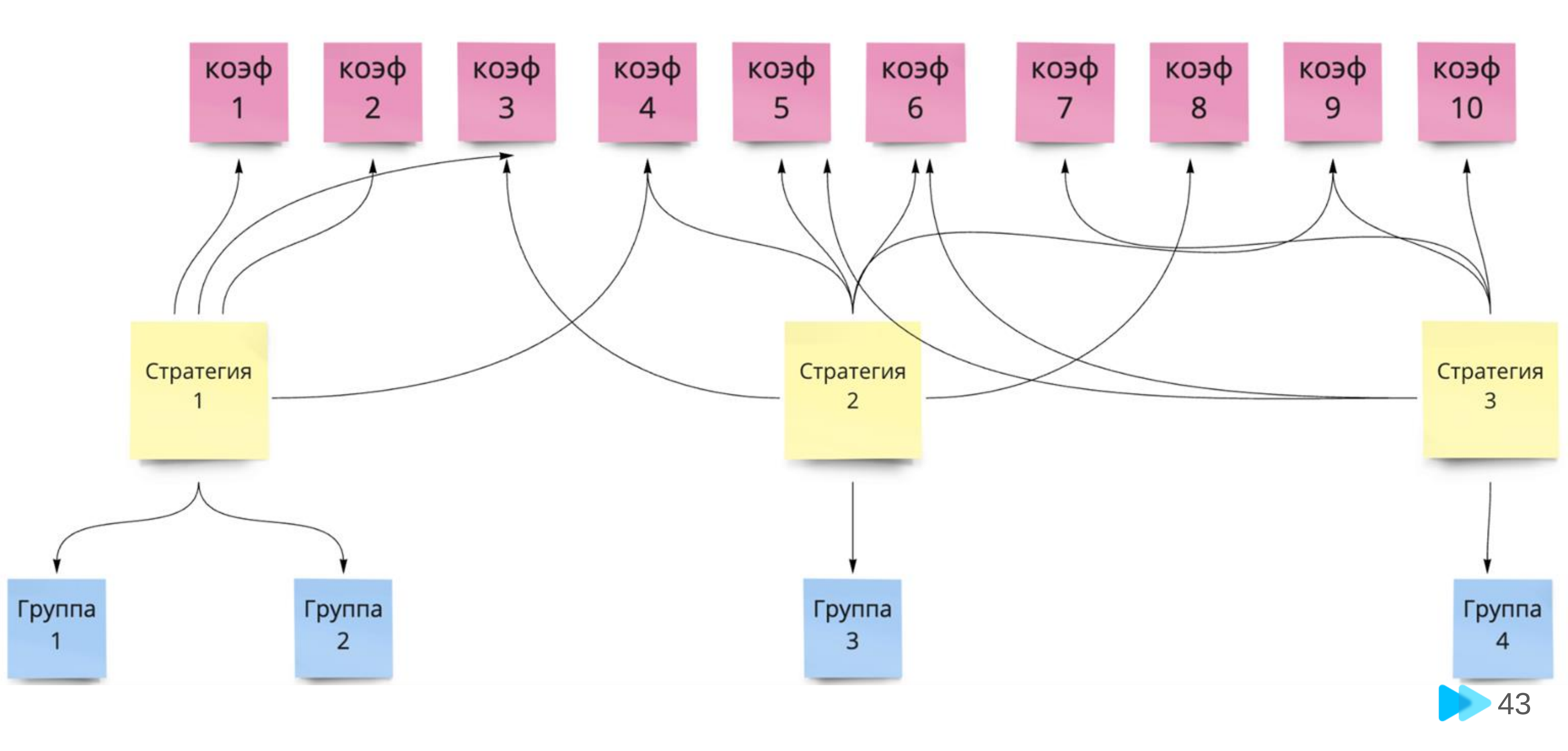
где Wn – полученный вес коэффициента.

- TaskWeightCalculator — Корень компонента
  - Coefficient
    - CoefficientInterface.php
    - ExampleCoefficient.php
    - HasIntroLessonCoefficient.php
    - HasLessonCoefficient.php
    - HasRegularScheduleCoefficient.php
    - NewTaskTaskTypeCoefficient.php
  - Strategy
    - AbstractClassicStrategy.php
    - AbstractTheoryOfGamesStrategy.php
    - ExampleStrategy.php
    - StrategyInterface.php
  - MissingCoefficientConfigurationException.php
  - TaskWeightCalculator.php — Расчёт приоритета, входная точка
  - TaskWeightCalculatorInterface.php
  - UnknownStrategyException.php

- TaskWeightCalculator — Корень компонента
  - Coefficient
    - CoefficientInterface.php
    - ExampleCoefficient.php
    - HasIntroLessonCoefficient.php
    - HasLessonCoefficient.php
    - HasRegularScheduleCoefficient.php
    - NewTaskTaskTypeCoefficient.php
  - Strategy — Стратегии расчёта
    - AbstractClassicStrategy.php — Расчёт по “старой” формуле
    - AbstractTheoryOfGamesStrategy.php — Расчёт, основанный на дереве игры
    - ExampleStrategy.php — Конкретная стретегия
    - StrategyInterface.php
  - MissingCoefficientConfigurationException.php
  - TaskWeightCalculator.php — Расчёт приоритета, входная точка
  - TaskWeightCalculatorInterface.php
  - UnknownStrategyException.php

- TaskWeightCalculator — Корень компонента
  - Coefficient
    - CoefficientInterface.php
    - ExampleCoefficient.php — Конкретный коэффициент
    - HasIntroLessonCoefficient.php
    - HasLessonCoefficient.php
    - HasRegularScheduleCoefficient.php
    - NewTaskTaskTypeCoefficient.php
  - Strategy — Стратегии расчёта
    - AbstractClassicStrategy.php — Расчёт по “старой” формуле
    - AbstractTheoryOfGamesStrategy.php — Расчёт, основанный на дереве игры
    - ExampleStrategy.php — Конкретная стретегия
    - StrategyInterface.php
  - MissingCoefficientConfigurationException.php
  - TaskWeightCalculator.php — Расчёт приоритета, входная точка
  - TaskWeightCalculatorInterface.php
  - UnknownStrategyException.php

Технические детали. Что представляет собой стратегия



```php
final class TaskWeightCalculator implements TaskWeightCalculatorInterface
{
    public function __construct(
        private iterable $strategies,
        private LoggerInterface $logger
    ) {
        $this->strategies = [...$strategies];
    }

    public function calculate(Task $task): int {...}
    private function resolveStrategy(Task $task): StrategyInterface {...}
}
```

```php
private function resolveStrategy(Task $task): StrategyInterface
{
    foreach ($this->strategies as $strategy) {
        if ($strategy->supports($task->getOperatorGroup())) {
            return $strategy;
        }
    }

    throw new UnknownStrategyException($task);
}
```

```php
public function calculate(Task $task): int
{
    try {
        $strategy = $this->resolveStrategy($task);
    } catch (UnknownStrategyException $exception) {
        $this->logger->error($exception->getMessage());

        return 0;
    }

    $weight = $strategy->calculate($task);

    return (int) $weight;
}
```

```php
abstract class AbstractTheoryOfGamesStrategy implements StrategyInterface
{
    public function calculate(Task $task): float
    {
        $summaryCoefficients = $this->getSummaryCoefficients();
        $summaryWeights = array_map(
            static fn (CoefficientInterface $coefficient) => $coefficient->calculate($task),
            $summaryCoefficients
        );

        return array_sum($summaryWeights);
    }
    abstract protected function getSummaryCoefficients(): array;
```

```php
final class ExampleStrategy extends AbstractTheoryOfGamesStrategy
{
    public function __construct(...) {...}

    public function supports(OperatorGroup $operatorGroup): bool
    {
        return $operatorGroup->hasOption(OptionEnum::STRATEGY_OPTION);
    }

    protected function getSummaryCoefficients(): array
    {
        return $this->coefficients;
    }
}
```

```php
public function __construct(
    HasIntroLessonCoefficient $hasIntroLessonCoefficient,
    HasRegularScheduleCoefficient $hasRegularScheduleCoefficient,
    NewTaskTaskTypeCoefficient $newTaskTaskTypeCoefficient,
)
{
    $this->coefficients = [
        $hasIntroLessonCoefficient->withNoIntroLessonWeight(1),
        $hasRegularScheduleCoefficient->withLessonStartedWeight(1),
        $newTaskTaskTypeCoefficient->withWeightMap([
            Type::CALL_BACK_TO_STUDENT => 1
        ]),
    ];
}
```

```
final class ExampleCoefficient implements CoefficientInterface
{
    private int $weight = 1;
    private int $defaultWeight = 0;

    public function withWeight(int $weight): self {...}  – Установка веса

    public function calculate(Task $task): float {...}  – Расчет приоритета
}
```

```php
public function calculate(Task $task): float
{
   $educationService = $this->crmApiClient->getEducationServicesByStudentId(
         $task->getStudentId()
   );
   if ($educationService === []) {
       return $this->defaultWeight;
   }

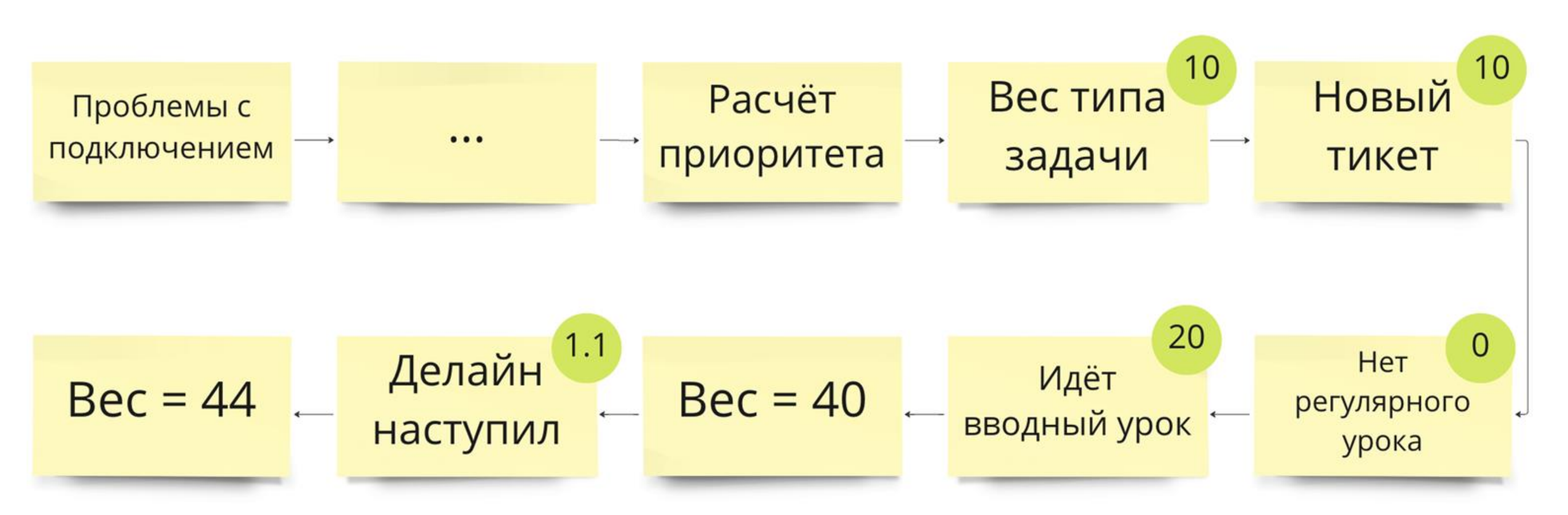
   return $this->weight;
}
```

Добавляем умножающие коэффициенты

Вес задачи = (W1 + W2 + … + W3) * W4 * … * Wn,

где Wn – полученный вес коэффициента.

Добавляем умножающие коэффициенты



```php
/* Суммирующие нструменты */
$this->summanryCoefficients = [
    $hasIntroLessonCoefficient->withNoIntroLessonWeight(1),
    $hasRegularScheduleCoefficient->withLessonStartedWeight(1),
    $newTaskTaskTypeCoefficient->withWeightMap([
        Type::CALL_BACK_TO_STUDENT => 1
    ]),
];


/* Умножающие нструменты */
$this->multiplicationCoefficients = [
    $deadlineCoefficient,
];
```

```php
$weight = array_sum($summaryWeights);

return array_reduce(
    $multiplicationWeights,
    function ($multiplication, $result): float
       $result * $multiplication,
    $weight
);
```

```php
foreach ($tasks as $task) {
    $task->getTaskDetails()->setExtraDeadlineRegistered(true);
    $this->taskRepository->save($task);

    $this->eventDispatcher->dispatch(new TaskOverdueEvent($task));
}
```

Что дальше?

- Изменение стратегии через разработку
- Клиенты ждут
- Польза доставляется долго

Урок сейчас

Тип класса	Собственный вес	Вес по умолчанию	Вес отсутствия урока
Суммирование	15	15	1

Вводный урок сейчас

Тип класса	Текущий урок - вводный	Текущий урок - не вводный	Собственный вес
Суммирование	1	10	1

Причина открытия

Тип класса	Собственный вес
Суммирование	25

Вес БП новой задач

Название резолюции	Код резолюции	Вес
Новая задача	new	50
Перезвонить	callback	90
Требуется контроль запроса	task_control_is_needed	10

Итоги

- Анализ задач по параметрам
- Быстрая сборка стратегии для группы
- Тонкая настройка стратегии под потребности
- Автоматизировали анализ тысяч задач

Спасибо!
Голосуйте за доклад :-)



- ➔ https://t.me/mifleo
- ➔ https://github.com/olegmifle
- ➔ mifleov@gmail.com



PHP Russia 2022